Towards Autonomic Management of NASA Missions

C.A. Rouff[1] M.G. Hinchey[2] J.L. Rash[2] W.F. Truszkowski[2] R. Sterritt[3]

[1]SAIC
Advanced Concepts B.U.
McLean, VA, USA
rouffc@saic.com

[2]NASA Goddard Space Flight Center
Greenbelt, MD, USA
{michael.g.hinchey, james.l.rash,
walter.f.truszkowski}@nasa.gov

[3]School of Computing
University of Ulster
Northern Ireland
r.sterritt@ulster.ac.uk

Abstract

Increasingly, NASA will rely on autonomous systems concepts, not only in the mission control centers on the ground, but also on spacecraft and on rovers and other assets on extraterrestrial bodies to achieve the full range of advanced mission objectives. While autonomy cost-effectively supports mission goals, autonomicity supports survivability of remote missions, especially when human tending is not feasible. Analysis of two prototype NASA agent-based systems and of a proposed mission involving numerous cooperating spacecraft illustrates how autonomous and autonomic system concepts may be brought to bear on future space missions.

1. Introduction

With NASA's renewed commitment to outer space exploration, greater emphasis is being placed on both human and robotic exploration. Even when humans are involved in the exploration, human tending of assets becomes cost-prohibitive or is not feasible. In addition, certain exploration missions will require spacecraft that will be capable of venturing where humans cannot be sent.

Until the mid-1980s, all space missions were operated manually from ground control centers. The high costs of satellite operations prompted NASA and others to begin automating as many functions as possible. In our context, a system is autonomous if it can achieve its goals without human intervention. A number of more-or-less automated ground systems exist today, but work continues towards the goal of reducing operations costs to even lower levels. Cost reductions can be achieved in a number of areas. Greater autonomy of satellite ground control and spacecraft operations are two such areas. Autonomy is increasingly seen as a critical approach for all missions.

Though autonomy will be critical for future missions, it is also necessary that these missions have autonomic properties. Autonomy alone, absent autonomicity, will leave the spacecraft vulnerable to the harsh environment in which they have to work and most likely performance will degrade, or the spacecraft will be destroyed or will not be able to recover from faults. Ensuring that exploration spacecraft have autonomic properties will increase the survivability and therefore the likelihood of success of these missions.

NASA needs autonomicity in its future missions to ensure they can operate on their own to the maximum extent possible without human intervention or guidance. A case can be made that all of NASA's future systems should be autonomic, and exhibit the four key properties of autonomic systems: self-configuring, self-optimizing, self-healing and self-protecting [1,13]. The following discusses the need for each of these autonomic properties in NASA missions.

2. Overview of Two Agent-Based Systems

NASA GSFC has played a leading role in the development of agent-based approaches to realize NASA's autonomy goals. Agents in the Lights Out Ground Operations System (LOGOS) acted as surrogate human controllers and interfaced with legacy software that controllers normally used, and with humans. Based on the success of this first prototype, development began on ACT, an environment in which richer agent and agent-community concepts were developed through detailed prototypes and operational ground-based and space-based scenarios.

2.1 LOGOS

LOGOS is a proof-of-concept system consisting of a community of autonomous software agents that cooperate in order to perform functions previously performed by human operators who used traditional software tools such as orbit generators and command sequence planners. The agents were developed in Java and used an in-house software backplane for communication between the agents. The LOGOS community architecture is shown in Figure 1. LOGOS is made up of ten agents, some that interface with legacy software, some that perform services for the other agents in the community, and others that interface with an analyst or operator. All agents have the ability to communicate with all other agents in the

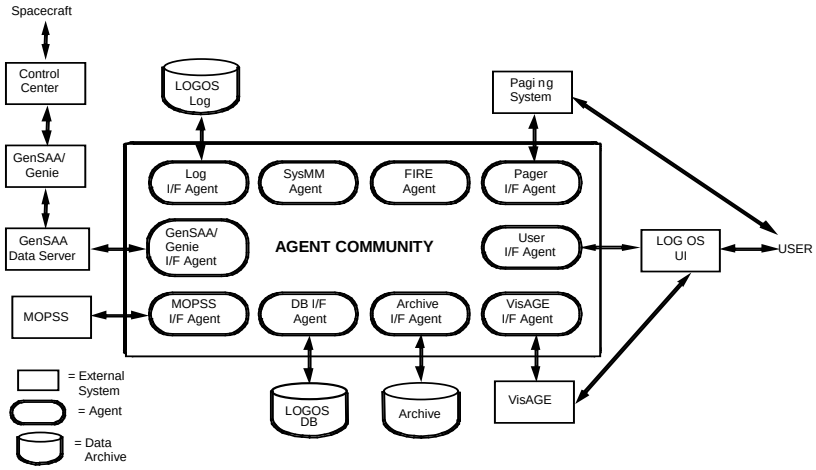

Fig. 1 LOGOS agent community and legacy software.

community. A more detailed description of LOGOS is given in [12].

The System Monitoring and Management Agent (SysMMA) maintains a list of all agents and their addresses in the community and provides their addresses to other agents requesting services. When started, each agent registers its capabilities with SysMMA and requests other agent addresses whose services it needs.

2.2 ACT

The motivation behind ACT was to develop a more flexible architecture than LOGOS for implementing a wide range of intelligent or reactive agents. After developing ACT, sample agents were built to simulate ground control of a satellite constellation mission as a proof of concept. Agents in ACT are built using a component architecture, where a component can be easily swapped in and out for easy removal of unneeded components for reactive agents and the inclusion of the necessary components to implement intelligent agents. It also allows for new technologies to be added as they become available without affecting previously implemented components. A simple (reactive) agent can be designed by using a minimum number of components that receive percepts (inputs) from the environment and react according to those percepts. A robust agent may use more complex components that allow the agent to reason in a deliberative, reflexive, and/or social fashion. The following are the components of ACT.

Modeler: The modeling component maintains the domain model of an agent, which includes models of the environment, other agents, and the agent itself. The Modeler is also responsible for reasoning with the models to act proactively and reactively with the environment and events that affect the model's state.

Reasoner: The Reasoner component works with information in its local knowledge base as well as model and state information from the Modeler to make decisions and formulate goals for the agent. Currently, the Reasoner works more in a reactive manner.

Planner/Scheduler: The Planner/Scheduler component is responsible for any agent-level planning and scheduling. The planning component receives a goal or set of goals to fulfill in the form of a plan request. This typically comes from the Reasoner component, but may be generated by any component in the system.

Agenda/Executive: The Agenda and the Executive work together to execute the plans developed by the Planner/Scheduler. The Executive executes the steps it receives from the Agenda. If the preconditions are met, the action is executed. When execution finishes, the Executive evaluates the post-conditions, and generates a completion status for that step. The completion status is then returned to the Agenda.

Agent Communications: The agent communication component is responsible for sending and receiving messages to/from other agents. The component takes an agent data object that needs to be transmitted to another agent and converts it to a message format understandable by the receiving agent.

Perceptors/Effectors: The Perceptors are responsible for monitoring the environment for the agent. Any data received by the agent from the environment, other than agent-to-agent messages, enters through Perceptors. The Effector is responsible for effecting or sending output to the agent's environment. Any agent output data, other than agent-to-agent messages, leaves through Effectors.

Agent Framework: The framework provides the base functionality for the components as well as the inter-component communication facility. The framework allows components to be easily added and removed from the

agent while providing a standard communications interface and functionality across all components.

3. ANTS: A Concept Mission

The NASA Autonomous Nano-Technology Swarm (ANTS) mission [9,10] will be made up of swarms of autonomous pico-class (approximately 1kg) satellites that will explore the asteroid belt. There will be approximately 1,000 spacecraft involved in the mission. Approximately 80 percent of the spacecraft will be workers (or specialists) that will have a single instrument onboard. Some will be coordinators (called leaders) that will have rules that decide the types of asteroids and data the mission is interested in and will coordinate the efforts of the workers. The third type of spacecraft are messengers and will coordinate communications between the workers, leaders and Earth. Each worker spacecraft will examine asteroids it encounters and send messages back to a coordinator that evaluates the data and sends other spacecraft to the asteroid if necessary.

This mission will involve a high degree of autonomy, and autonomic properties will enhance its survivability. To implement this mission a heuristic approach is being considered that uses an insect analogy of hierarchical social structure based on the above spacecraft hierarchy. A transport ship will assemble the spacecraft during the journey to the asteroid belt and then release them upon arrival. Sub-swarms will exist that will act as teams that explore a particular asteroid based on the asteroid's characteristics.

Team leaders contain models of the types of science they want to perform. Relevant goals are communicated to the messenger spacecraft that then relay them on to the worker spacecraft. The worker spacecraft then take measurements of asteroids using their specialized instrument until data matches the goal that was sent by the leader. If the data matches the profile of the type of asteroid that is being searched for, an imaging spacecraft will be sent to the asteroid to ascertain the exact location and to create a rough model prior to the arrival of other spacecraft.

4. Autonomic Properties of LOGOS

Self-configuration. LOGOS self configures when the GIFA agent receives signals from the GenSAA/Genie ground station software that a spacecraft pass is about to happen. When this occurs, the GenSAA/Genie Interface Agent (GIFA) configures the system by waking up the needed agents for the pass. For example, if there are no anomalies, then the Fault Isolation and REsolution (FIRE) agent is not needed and is not woken up. The same is true for the visualization and the user interface agents. If there is no user logged on, then those agents do not have to be woken up for the spacecraft pass.

Self-optimization. LOGOS self-optimizes through learning. One example of this is through the learning by the FIRE agent when it does not know how to resolve an anomaly, and must notify an analyst that it needs help. After the analyst provides a set of commands to resolve the anomaly, the FIRE agent stores those commands and the anomaly in its knowledge base for future reference. A second way that LOGOS self-optimizes is through the user interface and visualization agents. These agents keep track of which analyst looks at what data so that that information would be pre-fetched and available to the analyst when he or she logs on to the system.

Self-healing. LOGOS self-heals primarily through the actions of the FIRE agent. The FIRE agent examines anomalies that occur and then issues commands to resolve/heal the anomalies based on its knowledge base. It also self-heals through the intervention of the human in the loop, who can fill in information when the FIRE agent does not have the requisite knowledge to solve a problem. The self-healing aspect of LOGOS was its primary function and is what made the system lights-out.

Self-protecting. The self-protecting aspects of LOGOS are limited. The self-protection is primarily performed by the FIRE agent and the user interface agent (UIFA). UIFA accomplilshes self-protection when it authenticates a user logging on to the system. For the FIRE agent, self-protection is accomplished when checking commands entered by the analyst to ensure they do not harm the spacecraft.

5. Autonomic Properties of ACT

Self-configuration. As an example of this property, when ACT detects, from analysis of telemetry, that there is a problem, the Contact Manager alters the current satellite contact schedule to enable the problem to be addressed. What is being reconfigured, in this case, is the spacecraft functionality for managing communications contacts with ground systems and controllers.

Self-optimization. As an example of this property, consider what happens when a Proxy Agent determines that a problem exists with its spacecraft. When this situation arises, a replanning/rescheduling activity occurs to optimize the behavior of the entire ACT system.

Self-healing. Consider what happens when a Proxy Agent detects a problem with its associated spacecraft. Following a diagnosis of the problem (which may involve access to the human component of the ACT) corrective actions, in the form of commands, are generated and made ready for transmission to the affected spacecraft. This problem-diagnosis/corrective-action cycle is a major part of ACT's self-healing capability.

It should be noted that the three autonomic responses discussed above all stem from ACT's determination that a problem has occurred. In attending to the problem, ACT reconfigures, tries to optimize its operations, and proceeds to diagnose and solve the identified problem.

Self-protection. ACT is self-protecting in the sense that it constantly monitors the spacecraft systems and modifies its operations if a parameter ranges outside its normal bounds. In addition, it also has self-protection through validation of system commands to insure that command sequences executed will not harm the spacecraft or put it in a position where it could be harmed.

6. Autonomic Properties of ANTS

Self-configuration. As asteroids of interest are identified, appropriate teams of spacecraft are configured to realize optimal science operations at the asteroids. When operations are completed, the team disperses for reconfiguration at another asteroid. Reconfiguring may also be required as the result of a failure or anomaly. The loss of a given worker may result in the role of that worker being performed by another. Loss of communication with a worker may mean that the system has to assume loss of the worker, and the role may be allocated to another spacecraft. Loss of use of an instrument by a worker may require the worker to take the role of a communication device.

Self-optimization. Optimization of ANTS is accomplished at the individual level as well as at the system level. These optimizations are:

- Rulers learning about asteroids
- Messengers adjusting their position
- Workers learning about asteroids

Optimization at the ruler level is primarily through learning. Over time rulers will be collecting data on different types of asteroids and will learn the characteristics of the types of the asteroids that are of interest and the types of asteroids that are difficult to orbit or observe. From this information, the system as a whole is being optimized since time is not being wasted on asteroids that are not of interest.

Optimization for messengers is achieved through positioning. Messengers need to provide communications between the rulers and workers as well as back to Earth. This means that a messenger will have to be constantly adjusting its position to balance the communications between the rulers and workers and adjusting its position to send data to Earth.

Optimization at the worker level is primarily through experience gained with asteroids. As a worker observes asteroids and builds up a knowledge base of the different characteristics of asteroids, a worker may be able to automatically skip over asteroids that are not of interest.

Self-healing. The view of self-healing here is slightly different from that given in [1]. ANTS is self-healing not only in that it can recover from mistakes, but self-healing in that it can recover from failure, including damage from outside force. In the case of ANTS, these are non-malicious sources: events such as collision with an asteroid, or another satellite, loss of connection, etc., will require ANTS to heal itself by replacing one spacecraft with another.

ANTS mission self-healing scenarios span the range from negligible to severe. A negligible self-healing would be where one member of a redundant set of gamma ray sensors fails before a general gamma ray survey is planned. In such a scenario, the self-healing behavior would be the simple action of deleting the sensor from the list of functioning sensors. At the severe end of the range, an example scenario would arise when the team loses so many workers it can no longer conduct science operations. In this case, the self-healing behavior might be to advise mission control and wait for instructions. In some possible ANTS mission concepts, instead of "calling home" for help, an ANTS team may only need to request a replacement from another team.

ANTS individuals may also have self-healing behaviors. For example, an individual may have the capability of detecting corrupted code (software). In such a case, self-healing behavior would result in the individual requesting a copy of the affected software from another individual in the team, which would enable it to restore itself to a known operational state.

Self-protection. The self protecting behavior of the team will be interrelated with the self-protecting behavior of the individual members. The anticipated sources of threats to ANTS individuals (and consequently to the team itself) will be collisions and solar storms.

Collision avoidance through maneuvering will be limited because ANTS individuals will have limited ability to adjust their orbits and trajectories, since thrust for maneuvering is obtained from solar sails. Individuals will have to coordinate their orbits and trajectories with other individuals to avoid collisions. The main self-protection mechanism for collision avoidance is achieved through planning. The ruler's plans involve constraints that will result in acceptable risks of collisions between individuals.

Another possible ANTS self-protection mechanism could protect against effects of solar storms. Charged particles from solar storms could subject individuals to degradation of sensors and electronic components. When the ruler recognizes that a solar storm threat exists, the ruler would invoke its goal to protect the mission. In addition to its own protection, part of the ruler's response would be to give workers the goal to protect themselves.

As noted in the section on self-configuring behavior, after-effects of protective action will, in general,

necessitate ANTS self-reconfiguration. For example, after solar sails had been trimmed for the storm blast of solar wind, individuals will have unplanned trajectories, which will necessitate trajectory adjustments and replanning and perhaps new goals. Further, in case of the loss of individuals due to damage by charged particles, the ANTS self-healing behavior and the self-optimizing behavior may also be triggered. Thus, there is an interrelatedness of the self-protecting behaviors of the ANTS team and the ANTS individuals.

7. Conclusions

NASA missions represent some of the most extreme examples of the need for survivable systems that cannot rely on support and direction from humans while accomplishing complex objectives under dynamic and difficult conditions. Future missions will embody greater needs for longevity in the face of significant constraints, in terms of cost and the safety of human life. Future missions also will have increasing needs for autonomous behavior not only to reduce operations costs and overcome limitations of communications (signal propagation delays and low data rates), but also to overcome the limitations of humans to perform long-term space missions. There is an increasing realization that future missions must be not only autonomous, but also exhibit the properties of autonomic systems for the survivability of both individuals and systems.

As described, the LOGOS and ACT architectures provide for a flexible implementation of a wide range of intelligent and autonomic agents. The ACT architecture allows for easy removal of components unneeded for reactive agents, and the inclusion of the necessary components to implement intelligent and autonomic agents. It is also flexible so that additional unforeseen needs can be satisfied by new components that can be added without affecting previous components.

Acknowledgements

This work was supported in part by the NASA Office of Safety and Mission Assurance (OSMA) Software Assurance Research Program (SARP) and managed by the NASA Independent Verification and Validation (IV&V) Facility, and by NASA Headquarters Code R. And at the University of Ulster by the Computer Science Research Institute (CSRI) and the Centre for Software Process Technologies (CSPT) which is funded by Invest NI through the Centres of Excellence Programme, under the EU Peace II initiative.

References

[1] R. Murch, *Autonomic Computing*, IBM Press, 2004.

[2] C. Rouff, A. Vanderbilt, M. Hinchey, W. Truszkowski, and J. Rash. Properties of a Formal Method for Prediction of Emergent Behaviors in Swarm-based Systems. *2nd IEEE Int. Conf. Software Engineering and Formal Methods*. Beijing, China, 26-30 Sept., 2004.

[3] W. Truszkowski, and C. Rouff, An Overview of the NASA LOGOS and ACT Agent Communities, *5th World Multiconference on Systemics, Cybernetics, and Informatics*, Orlando, Florida, July 22-25, 2001.

[4] W. Truszkowski and H. Hallock, Agent technology from a NASA perspective. CIA-99, *3rd Int. Workshop on Cooperative Information Agents*, Springer-Verlag, Uppsala, Sweden, 31 July-2 August 1999.

[5] J. Ferber, *Multi-agent systems, An introduction to distributed artificial intelligence*. Addison-Wesley, 1999.

[6] M. Wooldridge, Intelligent Agents, in *Multiagent Systems*, Gerhard Weiss, Ed. MIT Press, 1999.

[7] P. Hughes, G. Shirah, and E. Luczak, Advancing Satellite Operations with Intelligent Graphical Monitoring Systems, *Proc. AIAA Computing in Aerospace Conference*, San Diego, CA, Oct. 19-21, 1993.

[8] W. Truszkowski and C. Rouff. A Process for Introducing Agent Technology into Space Missions.*Proc. IEEE Aerospace Conference*, March 11–16, 2001.

[9] P. E. Clark, S. A. Curtis, and M. L. Rilee, ANTS: Applying a New Paradigm to Lunar and Planetary Exploration, *Proc. Solar System Remote Sensing Symposium*, Pittsburg, 2002.

[10] S. A. Curtis, J. Mica, J. Nuth, G. Marr, M. Rilee, and M. Bhat, ANTS (Autonomous Nano-Technology Swarm): An Artificial Intelligence Approach to Asteroid Belt Resource Exploration, *Proc. International Astronautical Federation*, 51st Congress, October 2000.

[11] W. Truszkowski, J. Rash, C. Rouff and M. Hinchey, Asteroid Exploration with Autonomic Systems, *Proc. 11th IEEE Int. Conf. Engineering of Computer-Based Systems (ECBS), Workshop on Engineering of Autonomic Systems (EASe)*, Brno, Czech Republic, 24-27 May 2004, pp 484-489.

[12] W. Truszkowski, J. Rash, C. Rouff and M. Hinchey, Some Autonomic Properties of Two Legacy Multi-Agent Systems - LOGOS and ACT, *Proc. 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS), Workshop on Engineering of Autonomic Systems (EASe)*, Brno, Czech Republic, IEEE Computer Society Press, 24-27 May 2004, pp 490-498.

[13] C. Rouff, M. Hinchey, J. Rash, W. Truszkowski and R. Sterritt, Autonomic Properties of NASA Missions, *Proc. ICAC 2005, 2nd IEEE International Conference on Autonomic Computing*, Seattle, WA, 13-16 June 2005.